

# Inhaltsverzeichnis

<b>Vorwort</b> .....	13
<b>Geleitwort</b> .....	15
<b>Danksagungen</b> .....	21
<b>Einführung – Wie man dieses Buch lesen sollte</b> .....	23
<b>Teil I Wie Wandel funktioniert</b> .....	25
<b>1 Software ändern</b> .....	27
1.1 Vier Gründe, Software zu ändern .....	27
1.2 Riskante Änderungen .....	31
<b>2 Mit Feedback arbeiten</b> .....	33
2.1 Was sind Unit-Tests? .....	36
2.2 Higher-Level-Tests .....	39
2.3 Testabdeckung .....	39
2.4 Der Algorithmus zur Änderung von Legacy Code .....	42
<b>3 Überwachung und Trennung</b> .....	45
3.1 Kollaboratoren simulieren .....	47
<b>4 Das Seam-Modell</b> .....	53
4.1 Ein riesiges Blatt mit Text .....	53
4.2 Seams .....	54
4.3 Seam-Arten .....	57
<b>5 Tools</b> .....	69
5.1 Automatisierte Refactoring-Tools .....	69
5.2 Mock-Objekte .....	71
5.3 Unit-Test-Harnische .....	72
5.4 Allgemeine Test-Harnische .....	77

<b>Teil II Software ändern .....</b>	<b>79</b>
<b>6 Ich habe nicht viel Zeit und ich muss den Code ändern .....</b>	<b>81</b>
6.1 Sprout Method .....	83
6.2 Sprout Class .....	87
6.3 Wrap Method .....	91
6.4 Wrap Class .....	95
6.5 Zusammenfassung .....	100
<b>7 Änderungen brauchen eine Ewigkeit .....</b>	<b>101</b>
7.1 Verständlichkeit .....	101
7.2 Verzögerungszeit .....	102
7.3 Dependencies aufheben .....	103
7.4 Zusammenfassung .....	108
<b>8 Wie füge ich eine Funktion hinzu? .....</b>	<b>109</b>
8.1 Test-Driven Development (TDD) .....	110
8.2 Programming by Difference .....	116
8.3 Zusammenfassung .....	125
<b>9 Ich kann diese Klasse nicht in einen Test-Harnisch einfügen .....</b>	<b>127</b>
9.1 Der Fall des irritierenden Parameters .....	127
9.2 Der Fall der verborgenen Dependency .....	134
9.3 Der Fall der verketteten Konstruktionen .....	138
9.4 Der Fall der irritierenden globalen Dependency .....	140
9.5 Der Fall der schrecklichen Include-Dependencies .....	148
9.6 Der Fall der Zwiebel-Parameter .....	152
9.7 Der Fall des Alias-Parameters .....	154
<b>10 Ich kann diese Methode nicht in einem Test-Harnisch ausführen .....</b>	<b>159</b>
10.1 Der Fall der verborgenen Methode .....	159
10.2 Der Fall der »hilfreichen« Sprachfunktion .....	163
10.3 Der Fall des nicht erkennbaren Nebeneffekts .....	166
<b>11 Ich muss eine Änderung vornehmen. Welche Methoden sollte ich testen? .....</b>	<b>173</b>
11.1 Effekte analysieren .....	173
11.2 Vorwärtsanalyse (Reasoning Forward) .....	179
11.3 Effektfortpflanzung (Effect Propagation) .....	184
11.4 Tools für Effektanalysen .....	186
11.5 Von der Effektanalyse lernen .....	188
11.6 Effektskizzen vereinfachen .....	189

<b>12</b>	<b>Ich muss in einem Bereich vieles ändern. Muss ich die Dependencies für alle beteiligten Klassen aufheben? . . . . .</b>	193
12.1	Abfangpunkte . . . . .	194
12.2	Ein Design mit Einschnürpunkten beurteilen . . . . .	201
12.3	Fallen bei Einschnürpunkten . . . . .	203
<b>13</b>	<b>Ich muss etwas ändern, weiß aber nicht, welche Tests ich schreiben soll . . . . .</b>	205
13.1	Charakterisierungs-Tests . . . . .	206
13.2	Klassen charakterisieren . . . . .	209
13.3	Gezielt testen . . . . .	210
13.4	Eine Heuristik für das Schreiben von Charakterisierungs-Tests. . . . .	215
<b>14</b>	<b>Dependencies von Bibliotheken bringen mich um . . . . .</b>	217
<b>15</b>	<b>Meine Anwendung besteht nur aus API-Aufrufen. . . . .</b>	219
<b>16</b>	<b>Ich verstehe den Code nicht gut genug, um ihn zu ändern. . . . .</b>	227
16.1	Notizen/Skizzen . . . . .	228
16.2	Listing Markup . . . . .	229
16.3	Scratch Refactoring . . . . .	230
16.4	Ungenutzten Code löschen . . . . .	231
<b>17</b>	<b>Meine Anwendung hat keine Struktur . . . . .</b>	233
17.1	Die Geschichte des Systems erzählen . . . . .	234
17.2	Naked CRC . . . . .	238
17.3	Conversation Scrutiny . . . . .	241
<b>18</b>	<b>Der Test-Code ist im Weg. . . . .</b>	243
18.1	Konventionen für Klassennamen . . . . .	243
18.2	Der Speicherort für Tests . . . . .	244
<b>19</b>	<b>Mein Projekt ist nicht objektorientiert. Wie kann ich es sicher ändern? . . . . .</b>	247
19.1	Ein einfacher Fall . . . . .	248
19.2	Ein schwieriger Fall . . . . .	248
19.3	Neues Verhalten hinzufügen . . . . .	252
19.4	Die Objektorientierung nutzen . . . . .	255
19.5	Es ist alles objektorientiert . . . . .	258
<b>20</b>	<b>Diese Klasse ist zu groß und soll nicht noch größer werden. . . . .</b>	261
20.1	Aufgaben erkennen . . . . .	264
20.2	Andere Techniken . . . . .	278

## Inhaltsverzeichnis

20.3	Die nächsten Schritte .....	278
20.4	Nach dem Extrahieren von Klassen .....	281
21	<b>Ich ändere im ganzen System denselben Code.</b> .....	283
21.1	Erste Schritte .....	286
22	<b>Ich muss eine Monster-Methode ändern und kann keine Tests dafür schreiben</b> .....	301
22.1	Spielarten von Monstern.....	301
22.2	Monster mit automatischer Refactoring-Unterstützung zähmen .....	306
22.3	Die Herausforderung des manuellen Refactorings .....	308
22.4	Strategie .....	316
23	<b>Wie erkenne ich, dass ich nichts kaputtmache?</b> .....	319
23.1	Hyperaware Editing.....	319
23.2	Single-Goal Editing .....	321
23.3	Preserve Signatures .....	322
23.4	Lean on the Compiler .....	325
24	<b>Wir fühlen uns überwältigt. Es wird nicht besser.</b> .....	329
<b>Teil III Techniken zur Aufhebung von Dependencies</b> .....		333
25	<b>Techniken zur Aufhebung von Dependencies</b> .....	335
25.1	Adapt Parameter .....	335
25.2	Break Out Method Object.....	339
25.3	Definition Completion .....	345
25.4	Encapsulate Global References.....	347
25.5	Expose Static Method.....	353
25.6	Extract and Override Call.....	356
25.7	Extract and Override Factory Method.....	358
25.8	Extract and Override Getter.....	360
25.9	Extract Implementer .....	363
25.10	Extract Interface.....	368
25.11	Introduce Instance Delegator .....	374
25.12	Introduce Static Setter.....	376
25.13	Link Substitution .....	382
25.14	Parameterize Constructor .....	383
25.15	Parameterize Method .....	386

25.16	Primitivize Parameter . . . . .	388
25.17	Pull Up Feature . . . . .	390
25.18	Push Down Dependency. . . . .	394
25.19	Replace Function with Function Pointer. . . . .	397
25.20	Replace Global Reference with Getter . . . . .	400
25.21	Subclass and Override Method. . . . .	402
25.22	Supersede Instance Variable . . . . .	405
25.23	Template Redefinition. . . . .	409
25.24	Text Redefinition. . . . .	412
<b>A</b>	<b>Refactoring. . . . .</b>	<b>415</b>
A.1	Extract Method. . . . .	415
<b>B</b>	<b>Glossar . . . . .</b>	<b>421</b>
	<b>Stichwortverzeichnis. . . . .</b>	<b>423</b>